Energy Efficient Algorithms for the RFID Estimation Problem

Tao Li† Samuel Wu‡§ Shigang Chen† Mark Yang§
†Department of Computer & Information Science & Engineering
‡Department of Epidemiology and Health Policy Research
§Department of Statistics
University of Florida, Gainesville, FL 32611, USA

***Abstract*—RFID has been gaining popularity for inventory control, object tracking, and supply chain management in warehouses, retail stores, hospitals, etc. Periodically and automatically estimating the number of RFID tags deployed in a large area has many important applications in inventory management and theft detection. The prior work focuses on designing *time-efficient* algorithms that can estimate tens of thousands of tags in seconds. We observe that, for a RFID reader to access tags in a large area, active tags are likely to be used. These tags are battery-powered and use their own energy for information transmission. However, recharging batteries for tens of thousands of tags is laborious. Unlike the prior work, this paper studies the RFID estimation problem from the *energy angle*. Our goal is to reduce the amount of energy that is consumed by the tags during the estimation procedure. We design several energy-efficient probabilistic algorithms that iteratively refine a control parameter to optimize the information carried in the transmissions from the tags, such that both the number and the size of the transmissions are minimized.**

I. Introduction

The radio-frequency identification (RFID) technology has been widely used in various commercial applications, including inventory control, object tracking, and supply chain management. RFID tags (each storing a unique ID) are attached to goods at warehouses, merchandizes at retail stores, or equipment at hospitals, allowing an authenticated RFID reader to quickly access the properties of each individual item or collect the statistical information about a large group of items.

This paper focuses on a RFID-enabled function that is very useful in inventory management. Imagine a large warehouse storing thousands of refrigerators, tens of thousands of furniture pieces, or hundreds of thousands of footwear. A national retail survey showed that administration error, vendor fraud and employee theft caused about 20 billion dollars lost a year [1]. Hence, it is desirable to have a quick way of counting the number of items in the warehouse or in each section of the warehouse. To timely detect theft or management errors, such counting may be performed frequently.

If each item is attached with a RFID tag, the counting problem can be solved by a RFID reader that receives the IDs transmitted (or backscattered) from the tags [2]. However, reading the actual IDs of the tags can be time-consuming because so many of them have to be delivered in the same low-rate channel and collisions caused by simultaneous transmissions by different tags make the matter worse. To address this problem, Kodialam and Nandagopal [3], [4] showed that the reading time can be greatly reduced through probabilistic methods that *estimate the number of tags with an accuracy that can be arbitrarily set*. This is called the *RFID estimation problem*. The follow-up work by Qian et al. [5] significantly reduces the estimation time. It can be shown that even for the applications that require reading the actual IDs of all tags, estimating the number of tags as a pre-processing step will help make the main procedure of reading the IDs much more efficient [3]. Another advantage of estimating the number of tags without reading the IDs is that it ensures the anonymity of the tags, which may be useful in privacy-sensitive scenarios involving RFID-enhanced passports or driver's licences, where counting the number of people present is needed but revealing their identities is not necessary.

Is time efficiency the only performance metric for the RFID estimation problem? We argue that energy cost is also an important issue that must be carefully dealt with. In any application that requires a RFID reader to access tags in a large area, it is likely that battery-powered *active tags* will be used. *Passive tags* harvest energy from the radio signal of the reader and use such minute amount of energy to deliver information back to the reader. Their typical reading range is only several meters, which do not fit well with the big warehouse scenario. Active tags use their own power to transmit. A longer reading range can be achieved by transmitting at higher power. They are also richer in resources for implementing advanced functions. Their price becomes less of a concern if they are used for expensive merchandizes (such as refrigerators) or reused many times as goods moving in and out of the warehouse. But active tags also have a problem. They are powered by batteries. Recharging batteries for tens of thousands of tags is a laborious operation, considering that the tagged products may stack up, making tags not easily accessible. To prolong the tags' lifetime and reduce the frequency of battery recharge, all functions that involve large-scale transmission by many tags should be made energy-efficient. The prior work focuses on energy-efficient anti-collision protocols that minimize the energy consumption of a mobile reader [6], [7] when the reader collects the IDs of the tags. We believe this paper is the first to study energy-efficient solutions for the RFID estimation problem (which does not require reading the IDs of all tags).

The paper has three major contributions. First, we observe that there exists an asymmetry in energy cost. Solving the RFID estimation problem incurs energy cost both at the RFID reader and at the active tags. The asymmetry is that the energy cost at the tags should be minimized while the energy cost at the

reader is relatively a less concern because the reader's battery can be easily replaced or it may be powered by an external source. To exploit this asymmetry, our new algorithms follows a common framework that trades more energy cost at the reader for less cost at the tags. The reader will continuously refine and broadcast a control parameter called *contention probability*, which optimizes the amount of information the reader can extract from the transmissions by the tags. This in turn reduces the number of transmissions by the tags that are necessary to achieve a certain estimation accuracy.

Second, we investigate statistical methods, including the maximum likelihood estimation (MLE) and the average sum estimation (ASE) that are different from the probabilistic counting methods [8] used by [3], [4]. Our estimation algorithms optimize their performance by iteratively applying MLE or ASE with continuously refined parameters. The new algorithms not only require fewer transmissions by the tags but also minimize the size of each transmission. The number of transmissions made by the tags in our best algorithm is less than one third of the number in the state-of-the-art algorithms. In terms of the total number of bits transmitted by the tags, it is more than an order of magnitude smaller.

Third, we formally analyze the confidence intervals of the estimations made by the new algorithms and establish the termination conditions for any given accuracy requirement. We perform extensive simulations to demonstrate that the measured results match well with the analytical results and that the new algorithms perform far better in terms of energy saving than the best existing algorithms.

The rest of the paper is organized as follows. Section 2 discusses the related work. Section 3 defines the problem to be solved and the system model. Sections 4-6 propose three new solutions for the RFID estimation problem. Section 7 evaluates the new algorithms through simulations. Section 8 draws the conclusion.

II. Related Work

Most existing work focuses on how to efficiently read the IDs of the RFID tags. When queried by the RFID reader, the tags will respond with their IDs. Since the communication environment is wireless, inevitably collisions will happen when multiple tags choose the same slot. Collision arbitration protocols mainly fall into two categories: framed ALOHA-based approach [9], [10], [11], [7] and tree-based approach [12], [13], [14], [6]. In the former approach, the polling request carries the frame length, and each tag individually chooses a slot in the frame to transmit its ID. The process repeats until all the tags successfully transmit their IDs to the reader. In the later approach, the reader first sends out an ID prefix string. The tags whose IDs match the string will respond. If a collision happens, the reader will append a '0' or '1' to the prefix string and send out the new string. This process repeats until only one tag responds. Essentially the approach traverses a binary tree with the IDs of the tags being the leaf nodes.

Instead of identifying individual RFID tags, Kodialam and Nandagopal [3] study the problem of estimating the cardinality of a tag set. In their approach, information from tags are collected by a RFID reader in a series of time frames. Each frame consists of a number of slots, and tags probabilistically respond in those slots. Using the probabilistic counting methods, the reader estimates the number of tags based on the number of empty slots or the number of collision slots. Their best estimator is called the Unified Probabilistic Estimator (UPE). A follow-up work by the same authors proposes the Enhanced Zero-Based Estimator (EZB) [4], which makes its estimation based on the number of empty slots. The Lottery-Frame scheme (LoF) [5] by Qian et al. employs a geometric distribution-based scheme to determine which slot in a time frame each tag will respond. It significantly reduces the estimation time. However, every tag must respond during each of the time frames (their number ranges from tens to thousands), resulting in large energy cost when active tags use their own power to transmit. Also related is a novel security protocol proposed by Tan et al. to monitor the missing RFID tags in the presence of dishonest RFID readers [15].

None of the above estimators are designed with energy conservation in mind. In the following, we will present our energy efficient estimators.

III. Problem Definition And System Model

A. RFID Estimation Problem

The problem is to design efficient algorithms to estimate the number of RFID tags in a deployment area without actually reading the ID of each tag. Let N be the actual number of tags and $\hat{N}$ be the estimate. The estimation accuracy is specified by a confidence interval with two parameters: a probability value α and an error bound β, both in the range of $(0, 1)$. The requirement is that the probability for $\frac{N}{\hat{N}}$ to fall in the interval $[1-\beta, 1+\beta]$ is at least α, i.e.,

$$Prob((1-\beta)\hat{N} \leq N \leq (1+\beta)\hat{N}) \geq \alpha.$$

Our goal is to achieve the above estimation accuracy with minimum energy overhead to the active tags.

B. System Model

The type of active RFID systems considered in this paper is applicable to a large deployment area that are hundreds of feet or more across. Passive tags are beyond the scope of this paper. If they were used, one would have to take the reader and move around the whole area, collecting tag information once every few feet. Active tags allow a RFID reader to collect information from one location. The tagged goods (such as apparel) may stack in piles, and there may be obstacles, such as racks filled with merchandize, between a tag and the reader. We expect the active tags are designed to transmit with significant power that is high enough to ensure reliable information delivery in such a demanding environment. Hence, the energy cost due to the tags' transmissions is the main concern in our algorithm design; it increases at least in the square of the maximum distance to be covered by the RFID system. Energy consumption that powers the tag's circuit for computing and receiving information is not affected by long distance and obstacles. The energy consumed by the RFID reader is of less concern. We assume the reader transmits at sufficiently high power.

We use the following communication protocol between the reader and the tags. The reader first synchronizes the clocks of the tags and then performs a sequence of pollings. In each polling, the reader sends out a request, which is followed by a

slotted time frame during which the tags respond. The polling request from the reader carries a *contention probability* $0 < p \leq 1$ and a frame size f. Each tag will participate in the current polling with probability p. If it decides to participate, it will pick a slot uniformly at random from the frame, and transmit a bit string (called *response*) in that slot. The format of the response depends on the application. If the tag decides to not participate, it will keep silent. In our solutions, p will be set in the order of $\frac{1}{N}$.

If we know a lower bound N_{min} of N, the contention probability can be implemented efficiently to conserve energy. For example, a company's inventory of certain goods may be routinely in thousands and never be reduced below a certain number before, or the company has a policy on the minimum inventory, or the RFID estimation becomes unnecessary when the number of tags is below a threshold. In these cases, we will have a lower bound N_{min}, which can be much smaller than N. At the beginning of a polling, each tag makes a probabilistic decision: It goes to sleep for the whole polling period with probability $1 - \frac{1}{N_{min}}$ and wakes up until the next period starts, or it stays awake to receive the polling request with probability $\frac{1}{N_{min}}$ and then decides to respond with probability $p \times N_{min}$. For example, if $N = 10,000$ and $N_{min} = 1,000$, then only 10 tags stay awake in each polling. In Section IV-D, another energy-reduction method, called request-less pollings, will be proposed to eliminate most polling requests.

Optionally, the reader's request may include a predicate and only tags that satisfy the predicate will participate in the polling. For example, suppose all tags deployed in one section of a warehouse carry the 96-bit GEN2 IDs that begin with "000" in the Serial Number field. In order to estimate the number of tags in this section, the request carries a predicate testing whether the first three bits of a tag's Serial Number is "000".

A slot is said to be *empty* if no tag responds (transmits) in the slot. It is called a *singleton slot* if exactly one tag responds. It is a *collision slot* if more than one tag responds. A singleton or collision slot is also called a *non-empty slot*. The Philips I-Code system [16] requires a slot length of 10 bits in order to distinguish singleton slots from collision slots. On the contrary, one bit is enough if we only need to distinguish empty slots from non-empty slots — '0' means empty and '1' means non-empty. Hence, the response will be much shorter (or consume much less energy) if an algorithm only needs to know empty/non-empty slots, instead of all three types of slots as required by [3].

In order to prolong the lifetime of the tags, there are two ways to reduce their energy consumption: reducing the size of each response and reducing the number of responses. We will design algorithms that require only the knowledge of empty/non-empty slots and employ statistical methods to minimize the transmissions needed from the tags.

IV. Maximum Likelihood Estimation Algorithm

Our first estimator for the number of RFID tags is called the *maximum likelihood estimation algorithm* (MLEA). It fully utilizes the information from all pollings in order to minimize the number of tag responses it needs to meet the accuracy requirement.

A. Overview

MLEA uses the polling protocol described in Section III-B. The frame size f is fixed to be one slot. The reader adjusts the contention probability for each polling. Let p_i be the contention probability of the ith polling. MLEA only records whether the sole slot in each polling is empty or non-empty. Based on this information, it refines the estimate $\hat{N}$ until the accuracy requirement is met. Let z_i be the slot state of the ith polling. When at least one tag responds, the slot is non-empty and $z_i = 1$. When no tag responds, it is empty and $z_i = 0$. The sequence of z_i, $i \geq 1$, forms the *response vector*.

At the ith polling, each tag has a probability of p_i to transmit and, if any tag transmits, z_i will be one. Hence,

$$Prob\{z_i = 1\} = 1 - (1 - p_i)^N \approx 1 - e^{-Np_i}, \qquad (1)$$

where N is the the actual number of tags.

If the contention probabilities of the pollings are picked too small, the response vector will contain mostly zeros. If the contention probabilities are picked too large, the response vector will contain mostly ones. Both cases do not provide sufficient statistical information for accurate estimation. As will be discussed shortly, our analysis shows that the optimal contention probability is $p_i = 1.594/N$. The problem is that we do not know N (which is the quantity we want to estimate).

In order to determine p_i, MLEA consists of an *initialization phase* and an *iterative phase*. The former quickly produces a coarse estimation of N. The latter refines the contention probability and generates the estimation result.

B. Initialization Phase

We want to pick a small value for the initial contention probability. The expected number of responding tags at the first polling is Np_1. If p_1 is picked too large, a lot of tags will respond, which is wasteful because one response or many responses produce the same information — a non-empty slot. Suppose we know an upper bound N_{max} of N. This information is often available in practice. For example, we know N_{max} is 10,000 if the warehouse is designed to hold no more than 10,000 microwaves (each tagged with a RFID), or the company's inventory policy requires that in-store microwaves should not exceed 10,000, or the warehouse only has 10,000 RFID tags in use. N_{max} can be much bigger than N. We pick $p_1 = \frac{1}{N_{max}}$ such that the expected number of responding tags is no more than one. If $z_1 = 0$, we multiply the contention probability by a constant $C(> 1)$, i.e., $p_2 = p_1 \times C$ for the second polling. We continue multiplying the contention probability by C after each polling until a non-empty slot is observed. When that happens (say, at the lth polling), we have a coarse estimation of N to be $1/p_l$. Then we move to the next phase. When C is relatively large, the initialization phase only takes a few pollings to complete due to the exponential increase of the contention probability.

C. Iterative Phase

This phase iteratively refines the estimation result after each polling, and terminates when the specified accuracy requirement is met. Let $\hat{N}_i$ be the estimated number of tags after the ith polling. The reader performs three tasks. First, it sets the

contention probability as follows before sending out the polling request:

$$p_i = \frac{\omega}{\hat{N}_{i-1}}, \tag{2}$$

where $\hat{N}_{i-1}$ is the estimate after the previous polling and ω is a constant, which will be determined shortly. Second, based on the received z_i and the history information, the reader finds the new estimate of N that maximizes the following likelihood function:

$$L_i = \prod_{j=1}^{i} (1-p_j)^{N(1-z_j)}(1-(1-p_j)^N)^{z_j}, \tag{3}$$

where $(1-p_j)^{N(1-z_j)}(1-(1-p_j)^N)^{z_j}$ is the probability that the state of the slot in the jth polling is z_j. Namely, we want to find

$$\hat{N}_i = \arg\max_{N}\{L_i\}. \tag{4}$$

Third, after computing $\hat{N}_i$, the reader has to determine if the confidence interval of the estimation meets the requirement. In the following, we show how the above tasks can be achieved.

1) Determine the Contention Probability: Using the $\delta-$method [17], we derive the variance of $\hat{N}_i$ in Appendix A.

$$Var(\hat{N}_i) \approx \frac{1-(1-p_i)^N}{i(1-p_i)^N \ln^2(1-p_i)}. \tag{5}$$

When N is large and p_i is small, we can approximate $(1-p_i)^N$ as e^{-Np_i} and $\ln(1-p_i)$ as p_i. The above variance becomes

$$Var(\hat{N}_i) \approx \frac{e^{Np_i}-1}{ip_i^2}. \tag{6}$$

We pick the value of p_i that minimizes $Var(\hat{N}_i)$. Differentiating the right side of (6) and letting it be zero, we have $p_i = 1.594/N$. Because MLE approach provides statistically consistent estimate, when i is large, N can be closely approximated by $\hat{N}_{i-1}$. Hence, we set the value of p_i as follows:

$$p_i = \frac{1.594}{\hat{N}_{i-1}}. \tag{7}$$

2) Compute the value of $\hat{N}_i$: We compute the new estimate of N that maximizes (3). Since the maxima is not affected by monotone transformations, we use logarithm to turn the right side of the equation from product to summation:

$$\ln(L_i) = \sum_{j=1}^{i}\Big[N(1-z_j)\ln(1-p_j) + z_j\ln(1-(1-p_j)^N)\Big].$$

To find the maxima, we differentiate both sides:

$$\frac{\partial \ln(L_i)}{\partial N} = \sum_{j=1}^{i}\left[(1-z_j)\ln(1-p_j) - z_j\frac{(1-p_j)^N\ln(1-p_j)}{1-(1-p_j)^N}\right]. \tag{8}$$

We then set the right side to zero to solve for estimate $\hat{N}_i$. Note that the derivative is a monotone function of N, we can numerically obtain $\hat{N}_i$ through the bisection search.

3) Termination Condition: We show in Appendix A that, when i is large, $\hat{N}_i$ approximately follows the Gaussian distribution:

$$Norm\left(N, \frac{(1-(1-p_i)^N)}{i(1-p_i)^N\ln^2(1-p_i)}\right).$$

According to (6), the variance of $\hat{N}_i$ can be written as $\frac{e^{Np_i}-1}{ip_i^2}$. Hence, the confidence interval of N is

$$\hat{N}_i \pm Z_\alpha \cdot \sqrt{\frac{e^{\hat{N}_i p_i}-1}{ip_i^2}}, \tag{9}$$

where Z_α is the α percentile for the standard Gaussian distribution. For example, when $\alpha = 95\%$, $Z_\alpha = 1.96$. Because N is undetermined, we use $\hat{N}_i$ as an approximation when computing the standard deviation in (9).

The termination condition for EMLA is therefore

$$Z_\alpha \cdot \sqrt{\frac{e^{\hat{N}_i p_i}-1}{ip_i^2}} \le \hat{N}_i \cdot \beta, \tag{10}$$

where β is the error bound. The above inequality can be rewritten as

$$\sqrt{i} \ge \frac{Z_\alpha\sqrt{e^{\hat{N}_i p_i}-1}}{\hat{N}_i p_i \beta}. \tag{11}$$

Because $p_i = 1.594/\hat{N}_i$, we have

$$i \ge \frac{1.544 \cdot Z_\alpha^2}{\beta^2}. \tag{12}$$

Hence we can theoretically compute the approximate number of pollings that is required in order to meet the accuracy requirements. For example, if $\alpha = 95\%$ and $\beta = 5\%$, 2372 pollings will be required. Note that (12) is independent with the actual number of tags, N. Hence, our approach has perfect scalability.

Fig. 1 shows the simulation result of MLEA when $N = 10,000$, $\alpha = 95\%$ and $\beta = 5\%$. The simulation setup can be found in Section VII. The middle curve is the estimated number of tags, $\hat{N}_i$, with respect to the number pollings. It converges to the true value N represented by the central straight line. The upper and lower curves represent the 95% confidence interval, which shrinks as the number of pollings increases.

D. Request-less Pollings

We observe that, after a number of pollings, the value of p_i will stay in an increasingly small range and does not change much. It becomes unnecessary for the reader to transmit it at each polling. Hence, we improve MLEA as follows: If the percentage change in p_i during a certain number M of consecutive pollings is within a small threshold, the reader will broadcast the latest value of p_i and indicate that it will no longer transmit polling requests (unless the value of p_i changes beyond the threshold, in which case the reader will broadcast the new value of p_i one more time). Without receiving further polling requests, the tags will respond with the same contention probability in the subsequent slots. This is called the *request-less pollings*. Note that the termination condition in (10) remains correct. With the threshold being 10% and $M = 10$, the simulation results (which are omitted to save space) show

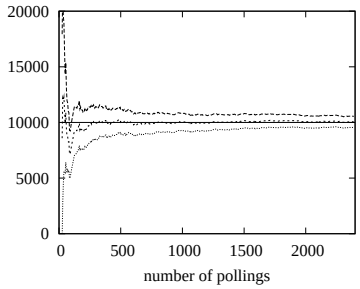

Fig. 1. The middle curve shows the estimated number of tags with respect to the number of pollings. The upper and lower curves show the confidence interval. The straightline shows the true number of tags.

that the performance difference caused by request-less pollings is negligibly small even though the contention probability during request-less pollings may be slightly off the value set by (7). Request-less pollings can also be applied to the algorithms in the next two sections.

E. Information Loss due to Collision

MLEA has a frame size of one slot. It obtains only binary information at each polling. Since the *target* contention probability is set to be $1.594/N$, the probability for more than one tag to respond in each polling is significant. However, no matter how many tags respond, the information that the reader receives is always the same, i.e., $z_i = 1$, which implies information loss when two or more tags decide to transmit at a polling. Let's compare two scenarios. In one scenario, only one tag responds at a polling. In the other, two tags respond. These two scenarios generate the same information but the energy cost of the second scenario is twice of the first. To address this problem, we design two more algorithms that reduce the probability of collision and, moreover, compensate the impact of collision in its computation.

V. Average Sum Estimation Algorithm

The *average sum estimation algorithm* (ASEA) is our second estimator for the number of RFID tags. It also utilizes history information from previous pollings. However, instead of only obtaining binary information, it computes the number of responses in each polling. Because more information can be extracted, it requires a fewer number of responses than MLEA. Moreover, the computation performed by ASEA after each polling is much simpler than MLEA's complicated bisection search.

A. Overview

ASEA uses the same polling protocol as MLEA does, except that its frame size f is larger than one in order to reduce the probability of collision. The result of the ith polling, x_i, is no longer a binary value. Instead, it is an estimate of the number of tags that respond at the polling.

ASEA takes three steps to solve the collision. First, it slightly reduces the contention probability p_i. Second, it increases the frame size f such that the tags that decide to respond at a polling are likely to respond at different slots in the frame. We pick values for p_i and f such that the collision probability is negligibly small. Third, we compensate the impact of collision in our computation.

ASEA also consists of an *initialization phase* and an *iterative phase*. The initialization phase of ASEA is the same as the initialization phase of MLEA, except that when the reader obtains the first non-zero result x_l at the lth polling with probability p_l, it computes a coarse estimation of N as $\frac{x_l}{p_l}$. Then it moves to the next phase, which is described below.

B. Iterative Phase

This phase iteratively refines the estimation after each polling, and terminates when the specified accuracy requirement is met. The reader performs four tasks in the ith iteration. First, it computes the contention probability p_i by (13) before sending out the polling request.

$$p_i = \frac{1}{\hat{N}_{i-1}}, \tag{13}$$

where $\hat{N}_{i-1}$ is the estimation after the previous polling. [1]

Second, the reader computes the number of responses x_i in the current frame.

Third, it computes $\hat{N}_i$ as follows:

$$\hat{N}_i = \frac{\sum_{j=l+1}^{i} x_j(1+\varepsilon)}{\sum_{j=l+1}^{i} p_j}, \tag{14}$$

where ε is introduced to compensate for collision and the iterative phase begins from the $(l+1)$th polling. The intuition behind (14) is given as follows: All pollings are performed independent of each other. Hence, the previous pollings with the contention probability $p_j (l < j \leq i)$ and the number of responses x_j can be treated as one polling with the contention probability $\sum_{j=l+1}^{i} p_j$ and the number of responses $\sum_{j=l+1}^{i} x_j$. The correctness of this treatment will be established by the analysis on the confidence interval and also by the simulation results.

Fourth, after computing $\hat{N}_i$, the reader determines if the estimate meets the accuracy requirement. In the following, we discuss the details of the second and fourth steps.

1) Compute the number of responses: At the ith polling, the reader measures the number of non-empty slots in the frame, denoted as x_i, which is an integer in the range of $[0..f]$. Due to possible collision, the actual number of responses, denoted as x_i^*, can be greater. However, we show that x_i closely approximates x_i^*. Their tiny difference can be computed and corrected.

Since each tag independently decides to respond with probability p_i, x_i^* follows a binomial distribution with parameters N and p_i, i.e.,

$$Prob\{x_i^* = k\} = \binom{N}{k} p_i^k (1-p_i)^{N-k}. \tag{15}$$

If $p_i \approx 1/N$ and N is sufficiently large, $Prob\{x_i^* = 2\} \approx 0.1839$, $Prob\{x_i^* = 3\} \approx 0.0613$, $Prob\{x_i^* = 4\} \approx 0.0153$, and the probability decreases exponentially with respect to k. $Prob\{x_i^* > 4\}$ is only about 0.0037. Next we compute the probability for collisions to happen at the ith polling, which is denoted as $Prob_i\{collision\}$.

[1] We could also use (7). In that case, the frame size f would increase accordingly to keep a small collision probability.

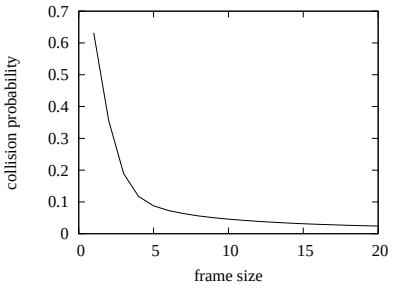

Fig. 2. The collision probability with respect to the frame size f.

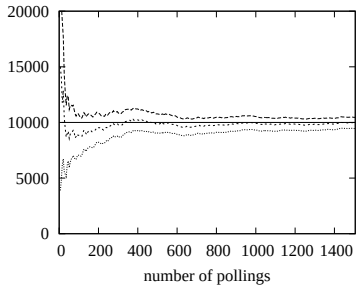

Fig. 3. The middle curve shows the estimated number of tags with respect to the number of pollings. The upper and lower curves show the confidence interval. The straightline shows the true number of tags.

$$Prob_i\{collision\} = \sum_{k=2}^{N} Prob_i\{collision|x_i^* = k\} \times Prob\{x_i^* = k\}$$

$$= \sum_{k=2}^{f}(1 - \frac{P(f,k)}{f^k}) \times Prob\{x_i^* = k\} + \sum_{k=f+1}^{N} 1 \times Prob\{x_i^* = k\}$$

where $P(f,k) = \frac{f!}{(f-k)!}$ is the permutation function. Fig. 2 shows the collision probability $Prob_i\{collision\}$ with respect to f. It diminishes quickly as f increases. When $f = 10$ (which is what we will choose in our simulations), $Prob_i\{collision\}$ is just 0.046. With such a small probability, the chance for more than two tags involved in a collision or more than one collision at one polling is exceedingly small and thus ignored. Therefore, to approximate x_i^*, we multiply x_i by 1.046 to compensate the impact of collision. Namely, $\varepsilon = 0.046$ in (14).

2) Termination Condition: Consider an arbitrary polling result, x_j, $j \leq i$. By our previous analysis, conditioned on p_j, we know that x_j^* has a binomial distribution $Bino(N, p_j)$ and that $(1+\varepsilon)x_j$ approximates x_j^*. Hence, when N is large enough, we can use the Gaussian distribution $Norm(\mu_j, \sigma_j)$ with parameters $\mu_j = Np_j$ and $\sigma_j = \sqrt{Np_j(1-p_j)}$ as a close approximation. Namely,

$$x_j^* \approx (1+\varepsilon)x_j \sim Norm(Np_j, Np_j(1-p_j)). \tag{16}$$

Consequently, $\sum_{j=l+1}^{i} x_j(1+\varepsilon)$ approximately follows Gaussian distribution

$$Norm\Big(N\sum_{j=l+1}^{i} p_j, N\sum_{j=l+1}^{i}(p_j(1-p_j))\Big). \tag{17}$$

Hence, the distribution of $\hat{N}_i = \sum_{j=l+1}^{i} x_j(1+\varepsilon)/\sum_{j=l+1}^{i} p_j$ can be approximated by

$$Norm\left(N, N\frac{\sum_{j=l+1}^{i}(p_j(1-p_j))}{(\sum_{j=l+1}^{i} p_j)^2}\right).$$

The confidence interval is

$$\hat{N}_i \pm Z_\alpha \cdot \sqrt{\frac{\hat{N}_i \sum_{j=l+1}^{i}(p_j(1-p_j))}{(\sum_{j=l+1}^{i} p_j)^2}}, \tag{18}$$

where Z_α is the α percentile for the standard Gaussian distribution. Hence, the termination condition is

$$Z_\alpha \cdot \sqrt{\frac{\hat{N}_i \sum_{j=l+1}^{i}(p_j(1-p_j))}{(\sum_{j=l+1}^{i} p_j)^2}} \leq \hat{N}_i \cdot \beta, \tag{19}$$

where β is the error bound.

Fig. 3 shows the simulation result of ASEA when $N = 10,000$, $\alpha = 95\%$ and $\beta = 5\%$. The middle curve is the value of $\hat{N}_i$, which converges to the value of N represented by the central straight line. The upper and lower curves represent the 95% confidence interval, which shrinks as the number of pollings increases. The algorithm terminates after 1517 pollings.

VI. Enhanced Maximum Likelihood Estimation Algorithm

This section presents our third estimator, the *enhanced maximum likelihood estimation algorithm* (EMLEA). EMLEA computes the number of responses x_i at each polling in the same way as ASEA does. But it uses the maximum likelihood method instead of the average sum method to estimate the number of tags. It is able to achieve the best energy efficiency among the three.

EMLEA uses the same polling protocol as ASEA does. It sets the same contention probability as ASEA does. It uses the same formula to estimate the number x_j^* of tags that respond in the jth polling. Namely $x_j^* \approx (1+\varepsilon)x_j$, where x_j is the number of non-empty slots in the frame and $\varepsilon = 0.046$ is used to compensate the impact of collision.

EMLEA differs from ASEA by using the maximum likelihood method instead of the average sum method to estimate the number of tags. Its termination condition is also different.

1) Compute the value of $\hat{N}_i$: Suppose the iterative phase starts at the $(l+1)$th polling. After the ith polling, the reader has collected the values of x_j, $l < j \leq i$. From (16), we know that the number of responding tags in the jth polling approximately follows a Gaussian distribution, $Norm(Np_j, Np_j(1-p_j))$. Hence, the probability for the *measured number of responding tags*, $(1+\varepsilon)x_j$, to occur under this distribution is $\frac{1}{\sqrt{2\pi Np_j(1-p_j)}} \cdot exp[-\frac{((1+\varepsilon)x_j - Np_j)^2}{2Np_j(1-p_j)}]$. The likelihood function for all measured numbers of responding tags in the pollings, $(1+\varepsilon)x_j$, $l < j \leq i$, to occur is

$$L_i = \prod_{j=l+1}^{i}\left[\frac{1}{\sqrt{2\pi Np_j(1-p_j)}} \cdot e^{-\frac{((1+\varepsilon)x_j - Np_j)^2}{2Np_j(1-p_j)}}\right]. \tag{20}$$

We want to find the value $\hat{N}_i$ that maximizes the likelihood function. Namely,

$$\hat{N}_i = \arg\max_{N}\{L_i\}. \tag{21}$$

We first take logarithm on both sides of (20).

$$\ln(L_i) = \sum_{j=l+1}^{i} \left[\ln \frac{1}{\sqrt{2\pi N p_j(1-p_j)}} - \frac{((1+\varepsilon)x_j - Np_j)^2}{2Np_j(1-p_j)} \right]. \tag{22}$$

We then differentiate both sides.

$$\begin{aligned} \frac{\partial \ln(L_i)}{\partial N} &= \sum_{j=l+1}^{i} \left[-\frac{1}{2N} + \frac{(1+\varepsilon)^2 x_j^2 - (Np_j)^2}{2N^2 p_j(1-p_j)} \right] \\ &= \sum_{j=l+1}^{i} \frac{(1+\varepsilon)^2 x_j^2 - (Np_j)^2}{2N^2 p_j(1-p_j)} - \frac{i-l}{2N}. \end{aligned} \tag{23}$$

Finally we set the right side to be zero and numerically compute the value of $\hat{N}_i$.

2) Termination Condition: The *fisher information*[2] $\mathcal{I}(\hat{N}_i)$ of L_i is defined as follows

$$\mathcal{I}(\hat{N}_i) = -E\left[\frac{\partial^2 \ln(L_i)}{\partial N^2} \right]. \tag{24}$$

According to (23), we have

$$\begin{aligned} \mathcal{I}(\hat{N}_i) &= E\left[\sum_{j=l+1}^{i} \frac{(1+\varepsilon)^2 x_j^2}{N^3 p_j(1-p_j)} - \frac{i-l}{2N^2} \right] \\ &= \sum_{j=l+1}^{i} \frac{(Np_j)^2 + Np_j(1-p_j)}{N^3 p_j(1-p_j)} - \frac{i-l}{2N^2} \\ &= \sum_{j=l+1}^{i} \frac{p_j}{N(1-p_j)} + \frac{i-l}{2N^2}. \end{aligned} \tag{25}$$

Above, we applied $E((1+\varepsilon)^2 x_j^2) = (Np_j)^2 + Np_j(1-p_j)$ in (25) since $(1+\varepsilon)x_j \sim Norm(Np_j, Np_j(1-p_j))$ and $E(x^2) = (E(x))^2 + Var(x)$.

Following the classical theory for MLE, when i is sufficiently large, the distribution of $\hat{N}_i$ is approximated by

$$Norm(N, \frac{1}{\mathcal{I}(\hat{N}_i)}). \tag{26}$$

Hence, the confidence interval is

$$\hat{N}_i \pm Z_\alpha \cdot \sqrt{\frac{1}{\mathcal{I}(\hat{N}_i)}}. \tag{27}$$

Note that we use $\hat{N}_i$ as an approximation for N in the computation when necessary since N is unknown. The termination condition for EMLEA to achieve the required accurary is

$$Z_\alpha \cdot \sqrt{\frac{1}{\mathcal{I}(\hat{N}_i)}} \leq \hat{N}_i \cdot \beta. \tag{28}$$

Fig. 4 shows the simulation result of EMLEA when $N = 10,000$, $\alpha = 95\%$ and $\beta = 5\%$. The middle curve is the value of $\hat{N}_i$, which converges to the value of N represented by the central straight line. The upper and lower curves represent the 95% confidence interval, which shrinks as the number of pollings increases. The algorithm terminates after 1081 pollings.

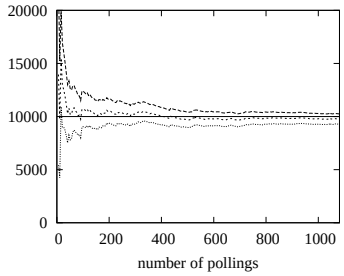

Fig. 4. The middle curve shows the estimated number of tags with respect to the number of pollings. The upper and lower curves show the confidence interval. The straightline shows the true number of tags.

TABLE I
NUMBER OF RESPONSES WHEN $\alpha = 90\%, \beta = 5\%$

N	Total number of responses					
	MLEA	ASEA	EMLEA	UPE-O	UPE-M	EZB
5000	2669 S	1083 S	768 S	7833 L	2415 L	10130 S
10000	2659 S	1083 S	763 S	13498 L	2432 L	20261 S
20000	2654 S	1081 S	775 S	24455 L	2431 L	40521 S

TABLE II
NUMBER OF RESPONSES WHEN $\alpha = 95\%, \beta = 5\%$

N	Total number of responses					
	MLEA	ASEA	EMLEA	UPE-O	UPE-M	EZB
5000	3798 S	1543 S	1083 S	8740 L	3437 L	14472 S
10000	3776 S	1543 S	1101 S	14573 L	3507 L	28944 S
20000	3789 S	1545 S	1099 S	25515 L	3450 L	57888 S

TABLE III
NUMBER OF RESPONSES WHEN $\alpha = 99\%, \beta = 5\%$

N	Total number of responses					
	MLEA	ASEA	EMLEA	UPE-O	UPE-M	EZB
5000	6542 S	2658 S	1857 S	11262 L	5885 L	24602 S
10000	6546 S	2654 S	1862 S	16969 L	5935 L	49205 S
20000	6543 S	2655 S	1887 S	28379 L	5939 L	98409 S

TABLE IV
NUMBER OF RESPONSES WHEN $\alpha = 99\%, \beta = 9\%$

N	Total number of responses				
	MLEA	ASEA	EMLEA	UPE-M	EZB
5000	2005 S	815 S	557 S	1802 L	7236 S
10000	2083 S	839 S	551 S	1929 L	14472 S
20000	2027 S	819 S	556 S	1755 L	28944 S

TABLE V
NUMBER OF RESPONSES WHEN $\alpha = 99\%, \beta = 6\%$

N	Total number of responses				
	MLEA	ASEA	EMLEA	UPE-M	EZB
5000	4766 S	1835 S	1260 S	3942 L	17366 S
10000	4756 S	1833 S	1240 S	4063 L	34733 S
20000	4775 S	1852 S	1227 S	3993 L	69465 S

TABLE VI
NUMBER OF RESPONSES WHEN $\alpha = 99\%, \beta = 3\%$

N	Total number of responses				
	MLEA	ASEA	EMLEA	UPE-M	EZB
5000	16263 S	7308 S	5014 S	15924 L	65124 S
10000	16211 S	7345 S	4942 S	16830 L	130247 S
20000	16291 S	7376 S	5055 S	16647 L	260495 S

VII. Simulation Result

In this section, we evaluate the performance of MLEA, ASEA and EMLEA by simulations. We compare the proposed algorithms with the state-of-the-art algorithms in the related work. They are the Unified Probabilistic Estimator (UPE) [3] and the Enhanced Zero-Based (ZEB) estimator [4]. The original UPE, denoted as UPE-O, is very energy-inefficient because its contention probability begins from 100% and thus all tags will respond. We modify it (denoted as UPE-M) to begin from a small initial contention probability $\frac{1}{N_{max}}$. We run each simulation 100 times and average the outcomes.

In the initialization phase of our protocols, let $N_{max} = 1,000,000$ and $C = 2$. The frame size in ASEA and EMLEA is 10 slots. The parameters for UPE and ZEB are picked based on the original papers whenever possible. All algorithms except for UPE need only to identify empty and non-empty slots. UPE has to identify empty, singleton and collision slots. To set a non-empty slot apart from an empty slot, a tag only needs to respond a short bit string to make the channel busy. However, to set a singleton slot apart from a collision slot, many more bits (10 used by UPE) are necessary [19]. For example, CRC may be used to detect collision.

The energy cost of an algorithm depends on (1) the number of responses that all tags transmit before the algorithm terminates and (2) the size of each response. We use 'S' to mean that the response is a short bit string (in the empty/non-empty case), and 'L' to mean a long bit string (in the empty/singleton/collision case).

We do not include the simulation results for LoF [5] because its energy cost is much higher than others. Its number of responses transmitted by the tags is kN, where k is the number of frames used in the estimation process.

A. Number of Responses

The first simulation studies the number of responses in each algorithm with respect to N, α and β. Table I shows the number of responses with respect to N when $\alpha = 90\%$ and $\beta = 5\%$. The proposed algorithms require fewer responses than UPE and EZB. As predicted, UPE-O is energy-inefficient; UPE-M works much better. The best algorithm is EMLEA, whose number of responses is about one third of what UPE-M requires and one fiftieth of what EZB when N is 20,000. Moveover, each response in UPE is much longer.

Tables II-III show similar comparison for different α values. Tables IV-VI show the comparison under different β values when $\alpha = 99\%$. We omit the results for UPE-O (which are much worse than the numbers of UPE-M) to keep the table width within a column. In all cases, the number of responses increases when α increases or β decreases, and except for EZB, the number does not vary much with respect to N, meaning that all algorithms except for EZB achieve good scalability. The ratio between the numbers for different algorithms appear to be quite stable under different parameter settings.

[2]The fisher information [18] is a way of measuring the amount of information that an observable random variable x carries about an unknown parameter θ upon which the likelihood function of θ, $L(\theta) = f(x;\theta)$, depends.

B. Total Number of Bits Transmitted

The second simulation evaluates the energy cost of the algorithms. As mentioned before, one bit is enough to separate empty/non-empty slot. Hence, the response of MLEA, ASEA, EMLEA and EZB is one bit long. A response in UPE-M is 10 bits long [3]. We compare the total number of bits transmitted by all tags before each algorithm terminates. Fig. 5 show the simulation results with respect to N when $\alpha = 95\%$ and $\beta = 3\%, 6\%$ and 9%. For example, when $\alpha = 95\%$, $\beta = 3\%$, and $N = 20,000$, the ratio between the number of bits transmitted by UPE-M (EZB) and that by our best estimator EMLEA is 32.8 (53.6). It should be noted that the number of bits transmitted is not an accurate measurement of the energy cost because it ignores the energy spent to power up the radio and synchronize with the reader. However, combining the number of bits and the number of transmissions (in the previous subsection) still gives a good idea on how energy-efficient each algorithm is.

C. Estimation Time

The third simulation compares the time it takes for each algorithm to complete the estimation of N. Based on the specification of the Philips I-Code system [16], after the required waiting times (e.g., gap between transmissions) are included, it can be calculated that a reader needs $0.4ms$ to detect an empty slot, $0.8ms$ to detect a collision or a singleton slot, and $1ms$ to broadcast a polling request. Hence, MLEA, ASEA, EMLEA and EZB requires a slot length of $0.4ms$, while UPE-M requires a slot length of $0.8ms$. Recall that the contention probability takes the form of $\frac{\omega}{\hat{N}_i}$, where ω is a known constant. Thus the reader transmits $\hat{N}_i$ instead of the actual probability value in the polling requests. If we assume N_{max} is no more than a million, then 20 bits for $\hat{N}_i$ are sufficient. MLEA has a fixed frame size of one slot. ASEA and EMLEA have a fixed frame size of 10 slots. EZB and UPE-M also have pre-determined frame sizes. Let $\alpha = 95\%$ and $\beta = 5\%$. Fig. (6) shows the estimation times of the algorithms with respect to the number of tags in the deployment. The times grow very slowly as the number of tags increase, which suggests the algorithms all scale well. UPE-M takes the least amount of time, only 1.1 seconds, to estimate 20,000 tags, while the other algorithms take between 3.4 to 7.7 seconds. EMLEA takes 5.3 seconds. Even though the new algorithms take longer to complete, their estimation time is still small. We believe the extra time needed can be well justified for the large energy saving.

VIII. Conclusions

This paper proposes three new probabilistic algorithms for estimating the number of RFID tags in a region. We believe the algorithms are the first of its kind that targets at prolonging the lifetime of the active RFIDs. Their energy cost is far less than the state-of-the-art algorithms in the related work.

References

[1] R. Hollinger and J. Davis, "National Retail Security Survey," *http://diogenesllc.com/NRSS_2001.pdf*, 2001.

[2] R. Want, "An Introduction to RFID Technology," *Proc. IEEE PerCom*, Jan 2006.

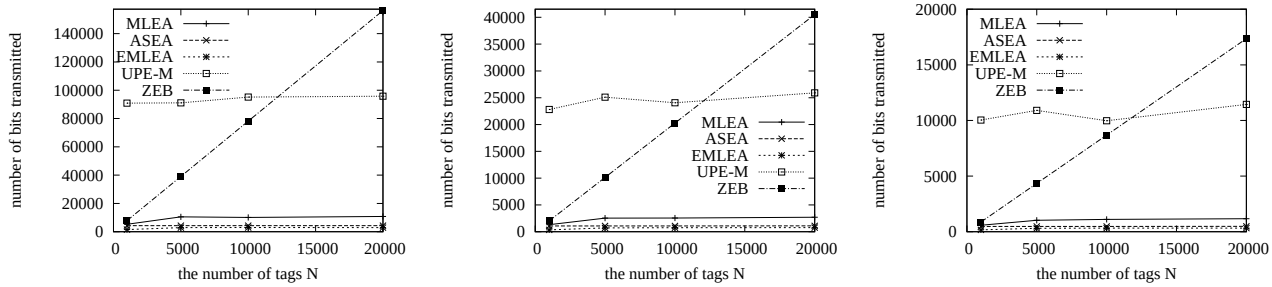

Fig. 5. $\alpha = 95\%, \beta = 3\%, 6\%$ and 9%.

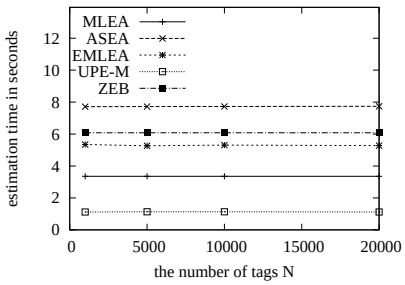

Fig. 6. Estimation times of the algorithms

[3] M. Kodialam and T. Nandagopal, "Fast and Reliable Estimation Schemes in RFID Systems," *Proc. ACM MOBICOM, Los Angeles*, 2006.

[4] M. Kodialam, T. Nandagopal, and W. Lau, "Anonymous Tracking using RFID tags," *Proc. IEEE INFOCOM*, 2007.

[5] C. Qian, H. Ngan, and Y. Liu, "Cardinality Estimation for Large-scale RFID Systems," *Proc. IEEE PerCom*, 2008.

[6] V. Namboodiri and L. Gao, "Energy-Aware Tag Anti-Collision Protocols for RFID Systems," *Proc. of IEEE PerCom*, 2007.

[7] D. Klair, K. Chin, and R. Raad, "On the Energy Consumption of Pure and Slotted Aloha based RFID Anti-Collision Protocols," *Computer Communications*, 2008.

[8] K. Hwang, B. Vander-Zanden, and H. Taylor, "A Linear-time Probabilistic Counting Algorithm for Database Applications," *ACM Transactions on Database Systems*, vol. 15, no. 2, June 1990.

[9] H. Vogt, "Efficient Object Identification with Passive RFID Tags," *Proc. IEEE PerCom*, 2002.

[10] J. Zhai and G. N. Wang, "An Anti-Collision Algorithm Using Two-functioned Estimation for RFID Tags," *Proc. ICCSA*, 2005.

[11] J. Cha and J. Kim, "Novel Anti-collision Algorithms for Fast Object Identification in RFID System," *Proc. IEEE ICPADS*, 2005.

[12] D. Hush and C. Wood, "Analysis of Tree Algorithm for RFID Arbitration," *Proc. IEEE ISIT*, 1998.

[13] J. Myung and W. Lee, "An adaptive memoryless tag anti-collision protocol for RFID networks," *Proc. IEEE ICC*, 2005.

[14] H. Choi, J. Cha, and J. Kim, "Fast Wireless Anti-collision Algorithm in Ubiquitous ID System," *Proc. IEEE VTC*, Sep 2004.

[15] Chiu C. Tan, Bo Sheng, and Qun Li, "How to monitor for missing RFID tags," *Proc. IEEE ICDCS*, June 2008.

[16] Philips Semiconductors, "I-CODE Smart Label RFID Tags," *http://www.nxp.com/acrobat_download/other/identification/SL092030.pdf*, Jan 2004.

[17] G. Casella and R. L. Berger, "Statistical Inference," *2nd edition, Duxbury Press*, 2002.

[18] Lehmann and G. Casella, "Theory of Point Estimation," *Springer, 2nd edition*, 1998.

[19] "EPC Radio-Frequency Identity Protocols Class-1 Generation-2 UHF RFID Protocol for Communications at 860 MHz - 960 MHz Version 1.0.9," *http://www.epcglobalinc.org/standards/uhfc1g2/uhfc1g2_1_0_9-standard-20050126.pdf*, EPCglobal, 2005.

APPENDIX A: DISTRIBUTION AND VARIANCE OF $\hat{N}_i$

Let i be a large positive integer. Consider the sequence of Bernoulli random variables, Z_j, $1 \leq j \leq i$, whose success probability is $q = 1-(1-p_i)^N$. Let $\hat{q} = (\sum_{j=1}^{i} Z_j)/i$, which is the estimation of the success probability q. It is known that asymptotically $\hat{q}$ follows a normal distribution:

$$\hat{q} \sim Norm\left(q, \frac{q(1-q)}{i}\right). \tag{29}$$

Because the MLE approach provides statistically consistent estimate, when i is large, we can consider the contention probabilities in the later stage of the pooling process to be approximatly a constant. In addition, the number of polling results before stabilization of the contention probability is limited, and their impact will diminish as i becomes large. That is, they can be ignored when the asymptotic property of $\hat{N}_i$ is considered. Hence, for the asymptotic property, we can let $p_j = p_i$, for $1 \leq j \leq i$, and Eq. (8) becomes

$$\frac{\partial \ln(L_i)}{\partial N} = \ln(1-p_i)\left[(i-\sum_{j=1}^{i} Z_j) - \frac{(1-p_i)^N}{1-(1-p_i)^N}\sum_{j=1}^{i} Z_j\right]. \tag{30}$$

Therefore, the MLE $\hat{N}_i$ that solves $\frac{\partial \ln(L_i)}{\partial N} = 0$ satisfies

$$(1-p_i)^{\hat{N}_i} = 1 - (\sum_{j=1}^{i} Z_j)/i = 1 - \hat{q}. \tag{31}$$

Hence, from (29), $(1-p_i)^{\hat{N}}$ asymptotically follows the following normal distribution

$$Norm\left((1-p_i)^N, \frac{(1-(1-p_i)^N)(1-p_i)^N}{i}\right). \tag{32}$$

According to the δ-method [17], if a random variable X_i satisfies

$$X_i \xrightarrow{D} Norm(\theta, \frac{\sigma^2}{i}), \tag{33}$$

where θ and σ are finite constants and $\xrightarrow{D}$ means convergence in distribution, then we must have

$$g(X_i) \xrightarrow{D} Norm\left(g(\theta), \frac{\sigma^2 [g'(\theta)]^2}{i}\right), \tag{34}$$

for any function g such that $g'(\theta)$ exists and takes a non-zero value. Based on (33) and (34), taking the logarithm of (32), we have

$$\hat{N}_i \cdot \ln(1-p_i) \sim Norm\left(N\ln(1-p_i), \frac{(1-(1-p_i)^N)}{i(1-p_i)^N}\right). \tag{35}$$

That is, $$\hat{N}_i \sim Norm\left(N, \frac{(1-(1-p_i)^N)}{i(1-p_i)^N \ln^2(1-p_i)}\right). \tag{36}$$

Hence, we have $$Var(\hat{N}_i) \approx \frac{(1-(1-p_i)^N)}{i(1-p_i)^N \ln^2(1-p_i)}.$$